SQUARE Process

Nancy R. Mead, Software Engineering Institute [vita[3]]

2006-01-30

System Quality Requirements Engineering (SQUARE) provides a means for eliciting, categorizing, and prioritizing security requirements for information technology systems and applications. The focus of the methodology is to build security concepts into the early stages of the development life cycle. The model can also be used for documenting and analyzing the security aspects of fielded systems and for steering future improvements and modifications to those systems.

Overview

Security Quality Requirements Engineering (SQUARE) is a model developed at Carnegie Mellon University by Nancy Mead as part of a research project with Donald Firesmith and Carol Woody of the Software Engineering Institute. This process provides a means for eliciting, categorizing, and prioritizing security requirements for information technology systems and applications. The focus of this methodology is to build security concepts into the early stages of the development life cycle. The model can also be used for documenting and analyzing the security aspects of fielded systems and for steering future improvements and modifications to those systems.

Subsequent to initial development, SQUARE was applied in a series of client case studies. Carnegie Mellon graduate students worked on this project during the summer and fall of 2004 and the summer of 2005. The case study results were published [Chen 04, Gordon 05, Xie 04]. Prototype tools were also developed to support the process. The draft process was revised based on the case studies; the revised process is shown in [. In principle, Steps 1-4 are actually activities that precede security requirements engineering but are necessary to ensure that it is successful. A detailed discussion of the method can be found in [Mead 05a].

Table 1. SQUARE Process

Number	Step	Input	Techniques	Participants	Output
1	Agree on definitions	Candidate definitions from IEEE and other standards	Structured interviews, focus group	Stakeholders, requirements team	Agreed-to definitions
2	Identify security goals	Definitions, candidate goals, business drivers, policies and procedures, examples	Facilitated work session, surveys, interviews	Stakeholders, requirements engineer	Goals
3	Develop artifacts to support security	Potential artifacts (e.g., scenarios,	Work session	Requirements engineer	Needed artifacts: scenarios,

3. daisy:230 (Mead, Nancy)

	requirements definition	misuse cases, templates, forms)			misuse cases, models, templates, forms
4	Perform risk assessment	Misuse cases, scenarios, security goals	Risk assessment method, analysis of anticipated risk against organizational risk tolerance, including threat analysis	Requirements engineer, risk expert, stakeholders	Risk assessment results
5	Select elicitation techniques	Goals, definitions, candidate techniques, expertise of stakeholders, organizational style, culture, level of security needed, cost/benefit analysis, etc.	Work session	Requirements engineer	Selected elicitation techniques
6	Elicit security requirements	Artifacts, risk assessment results, selected techniques	Joint Application Development (JAD), interviews, surveys, model-based analysis, checklists, lists of reusable requirements types, document reviews	Stakeholders facilitated by requirements engineer	Initial cut at security requirements
7	Categorize requirements as to level (system, software, etc.) and whether they are requirements or other kinds of constraints	Initial requirements, architecture	Work session using a standard set of categories	Requirements engineer, other specialists as needed	Categorized requirements

8	Prioritize requirements	Categorized requirements and risk assessment results	Prioritization methods such as Triage, Win-Win, etc.	Stakeholders facilitated by requirements engineer	Prioritized requirements
9	Requirements inspection	Prioritized requirements, candidate formal inspection technique	Inspection method such as Fagan, peer reviews, etc.	Inspection team	Initial selected requirements, documentation of decision-making process and rationale

How to Apply SQUARE

The SQUARE process is best applied by the project's requirements engineers and security experts, in the context of supportive executive management and stakeholders. We believe the process works best when elicitation occurs after risk assessment (Step 4) has been done and when security requirements are specified prior to critical architecture and design decisions. Thus critical business risks will be considered in the development of the security requirements.

Step 1, Agree on Definitions, is needed as a prerequisite to security requirements engineering. On a given project, team members will tend to have definitions in mind, based on their prior experience, but those definitions will not necessarily agree [Woody 05]. For example, to some government organizations, security has to do with access based on security clearance levels, whereas to others security may have to do with physical security or cyber security. It is not necessary to invent definitions. Most likely, sources such as IEEE and SWEBOK will provide a range of definitions to select from or tailor. A focus group meeting with the interested parties will most likely enable the selection of a consistent set of definitions for the security requirements activity.

Step 2, Identify Security Goals, should be done at the organizational level and is needed to develop the information system. This provides a consistency check with the organization's policies and operational security environment. Different stakeholders will likely have different goals. For example, a stakeholder in human resources may be concerned about maintaining the confidentiality of personnel records, whereas a stakeholder in a financial area may be concerned with ensuring that financial data is not accessed or modified without authorization. It is important to have a representative set of stakeholders, including those with operational expertise. Once the goals of the various stakeholders have been identified, they will need to be prioritized. In the absence of consensus, an executive decision may be needed to prioritize these goals.

Step 3, Develop Artifacts, is necessary to support all the subsequent activities. It is often the case that organizations do not have a documented concept of operations for a project, succinctly stated project goals, documented normal usage and threat scenarios, misuse cases, and other documents needed to support requirements definition. This means that either the entire requirements process is built on a foundation of sand or a lot of time is spent backtracking to try to obtain such documentation.

Step 4, Perform Risk Assessment, requires an expert in risk assessment methods, the support of the stakeholders, and the support of a requirements engineer. There are a number of risk assessment methods to select from. A specific method can be recommended by the risk assessment expert, based on the needs of the organization. The artifacts from Step 3 provide the input to the risk assessment process. The outcomes of the risk assessment can help in identifying the high-priority security exposures. Organizations that do not perform risk assessment typically do not have a logical approach to considering organizational risk when identifying security requirements but tend to select mechanisms,

such as encryption, without really understanding the problem that is being solved.

Step 5, Select Elicitation Technique, becomes important when there are several classes of stakeholders. A more formal elicitation technique, such as ARM [Hubbard 99], JAD [Wood 89], or structured interviews can be effective in overcoming communication issues when there are stakeholders with different cultural backgrounds. In other cases, elicitation may simply consist of sitting down with a primary stakeholder to try to understand that stakeholder's security requirements needs.

Step 6, Elicit Security Requirements, is the actual elicitation process using the selected technique. Most elicitation techniques provide detailed guidance on how to perform elicitation. This builds on the artifacts that were developed in earlier steps, such as misuse and abuse cases, attack trees, threats, and scenarios.

Step 7, Categorize Requirements, allows the requirements engineer to distinguish among essential requirements, goals (desired requirements), and architectural constraints that may be present. Requirements that are actually constraints typically occur when a specific system architecture has been chosen prior to the requirements process. This is good, as it allows assessment of the risks associated with these constraints. This categorization also helps in the prioritization activity that follows.

Step 8, Prioritize Requirements, depends not only on the prior step but may also involve performing a cost/benefit analysis to determine which security requirements have a high payoff relative to their cost.

Step 9, Requirements Inspection, can be done at varying levels of formality, from Fagan Inspections to peer reviews. Once inspection is complete, the organization should have an initial set of prioritized security requirements. It should also understand which areas are incomplete and must be revisited at a later time. Finally, the organization should understand which areas are dependent on specific architectures and implementations and should expect to revisit those as well.

How to Measure and Manage

Although quantitative measures do not exist, the clients for the case studies mentioned earlier recognized the value of the new security requirements and have started to take steps to incorporate them into the system. Important considerations for management are the amount of resources to be invested in this activity and in the implementation of the resultant requirements [Xie 04]. Management also needs to provide insights into the business environment and drivers and the mission of the system under development, as well as input as to the essential services and assets of the system.

Additional Resources for SQUARE

The following presentations and activities were intended to initiate further discussion on the management issues addressed by the SQUARE process:

- "Considering Operational Security Risks During Systems Development," SEPG 2004, Orlando, Florida, March 9, 2004 [Alberts 04]
- "Can Secure Systems be Built Using Today's Development Processes?" European SEPG, London, England, June 17, 2004 [Woody 04]

A workshop on Requirements for High Assurance Systems (RHAS '04) was held in conjunction with the International Conference on Requirements Engineering on September 6, 2004. The workshop proceedings for this and prior RHAS workshops were published by the SEI [SEI 04]. SQUARE was presented at an International Conference on Software Engineering (ICSE) workshop as well [Mead 05b]. The definitive technical report on SQUARE was published this year [Mead 05a].

Tools

A prototype tool called T-SQUARE has been developed to support SQUARE. It primarily provides an organizational framework for the artifact documents, and it also provides default content for some of the steps. It does not perform sophisticated functions such as requirements analysis. This prototype tool is undergoing further development in 2005-2006, so that it provides better support to the SQUARE process and is more attractive to users.

Results to Expect

When SQUARE is applied, the user should expect to have identified and documented relevant security requirements for the system or software that is being developed. SQUARE may be more suited to a system under development than one that has already been fielded, although it has been used both ways.

Relevant Metrics

There is no formal measurement data at this time, although clients have been satisfied that security requirements were identified that might not have been discovered otherwise and have taken steps to implement them.

Maturity of Practice

There have been several successful pilot projects using several versions of the SQUARE method while it has been under development. It is not a mature practice, however.

Case Studies

The SQUARE Methodology has undergone several case studies conducted by graduate students at Carnegie Mellon University [Chen 04, Gordon 05]. The goals of the case studies were to experiment with each step of the SQUARE process, make recommendations, and determine the feasibility of integrating SQUARE into standardized software development practices. The case studies involved real-world clients that were developing large-scale IT projects. The clients included an IT firm in Pittsburgh, Pennsylvania, a federal government research institute, and a department of the federal government.

Acme Corporation

All three case studies included Acme Corporation (Acme)[38], a private company headquartered in Pittsburgh. It provides technical and management services to various public sectors and a number of diversified private sectors. Its product under study, the Asset Management System (AMS)[39], provides a tool for companies to make strategic allocations and planning of their critical IT assets. It provides specialized decision support capabilities via customized views. AMS provides a graphical interface to track and analyze the state of important assets. The security requirements surrounding the AMS are the subject of these graduate case studies.

It is important to note here that the AMS is a fielded system, undergoing major upgrades, so the results

38. Acme Corporation (Acme) is an alias used to protect the identity of the client under study.

39. Asset Management System (AMS) is an alias used to protect the identity of the client under study.

from these case studies may not be a perfect fit for determining SQUARE's usefulness in a pre-production environment. However, the willingness of the client to participate was an important factor in its selection. Further, the results of these case studies are important in beginning to understand the effectiveness of the nine steps of the SQUARE process.

Output from SQUARE Steps

In each case study, the student teams focused part of their efforts on researching various methods to conduct each step. In some cases, redundant work was completed to determine which methods might lend themselves better to SQUARE. In order to provide concrete examples of the nine SQUARE steps, we present here a sample of the output from each individual step (all taken from the case studies) to demonstrate how SQUARE looks in action.

Step 1: Agree on Definitions

The student teams worked with the client to agree on a common set of security definitions in order to create a common base of understanding. The following is a minute subset of the definitions that were agreed upon:

- *access control*: Access control ensures that resources are granted only to those users who are entitled to them.
- *access control list*: A table that tells a computer operating system which access rights or explicit denials each user has to a particular system object, such as a file directory or individual file.
- *antivirus software*: A class of program that searches hard drives and floppy disks for any known or potential viruses.

The full set of definitions was drawn from resources such as Carnegie Mellon University, industry, and dictionaries.

Step 2: Identify Safety and Security Goals

Here, the project team worked with the client to flesh out safety and security goals that mapped to the company's overall business goal. The business and security goals were defined as follows:

- **Business Goal of AMS:** To provide an application that supports asset management and planning.
- **Safety and Security Goals:** Three high-level safety and security goals were derived for the system:
  1. Management shall exercise effective control over the system's configuration and usage.
  2. The confidentiality, accuracy, and integrity of the AMS shall be maintained.
  3. The AMS shall be available for use when needed.

Step 3: Developing Artifacts

Architectural diagrams, use cases, misuse cases, attack trees, and essential assets and services were documented in this portion of SQUARE. For instance, an attack scenario was documented in the following way:

System administrator accesses confidential information

1. by being recruited OR
   1. by being bribed OR

2. by being threatened OR
3. through social engineering OR

2. by purposefully abusing rights

This step creates a volume of important documentation that serves as vital input into following steps.

Step 4: Perform Risk Assessment

The risk management techniques that were field tested were selected after a literature review was completed. This literature review examined the usefulness and applicability of eight risk assessment techniques:

1. General Accounting Office Model [GAO 99]
2. National Institute of Standards Model [Stoneburner 02]
3. NSA's INFOSEC Assessment Methodology [NSA 04]
4. Shawn Butler's Security Attribute Evaluation Method [Butler 02]
5. Carnegie Mellon's Vendor Risk Assessment and Threat Evaluation [Lipson 01]
6. Yacov Haimes's Risk Filtering, Ranking, and Management Model [Haimes 04]
7. Carnegie Mellon's Survivable Systems Analysis Method [Mead 02]
8. Martin Feather's Defect Detection and Prevention Model [Cornford 04]

Each method was ranked in four categories:

1. suitability for small companies
2. feasibility of completion in the time allotted
3. lack of dependence on historical threat data
4. suitability in addressing requirements

After averaging scores from the four categories, NIST's and Haimes's models were selected as useful techniques for the risk assessment step. Many threat scenarios were brainstormed during this step. Some of this input came from the attack tree and misuse case documentation provided from Step 4. The two independent risk assessment analyses produced a useful risk profile for the company's system. The two most meaningful findings were

1. Insider threat poses the most important risk to the AMS.
2. Because of weak controls, it is easy for an insider or passerby to defeat authentication.

All findings from the risk assessment, along with the findings from the essential services and asset identification process completed in the artifact generation stage, were used to determine the priority level associated with each of the nine requirements.

Step 5: Select Elicitation Techniques

For this step, student teams were tasked with testing various elicitation techniques and models for the overall benefit of SQUARE. Although this task may appear to be straightforward, it is often the case that multiple techniques will likely work for the same project. The difficulty is in choosing a technique that can adapt to the number and expertise of stakeholders, the size and scope of the client project, and the

expertise of the requirements engineering team. It is extremely unlikely that any single technique will work for all projects under all circumstances, though previous experience has shown that the Accelerated Requirements Method (ARM) has been successful in eliciting security requirements.

The following is a sample of elicitation techniques that may be appropriate:

- structured/unstructured interviews
- use/misuse cases [Jacobson 92]
- facilitated meeting sessions, such as Joint Application Development and the Accelerated Requirements Method [Wood 89, Hubbard 99]
- Soft Systems Methodology [Checkland 89]
- Issue-Based Information Systems [Kunz 70]
- Quality Function Deployment [QFD 05]
- Feature-Oriented Domain Analysis [Kang 90]
- Controlled Requirements Expression [Mullery 79]
- Critical Discourse Analysis [Schiffrin 94]

Steps 6 and 7: Elicit and Categorize Safety and Security Requirements

Nine security requirements were derived and then organized to map to the three higher level security goals. Two of the nine requirements are

- Req 1: The system is required to have strong authentication measures in place at all system gateways/entrance points (maps to Goals 1 and 2).
- Req 3: It is required that a continuity of operations plan (COOP) be in place to assure system availability (maps to Goal 3).

The nine security requirements made up the heart of the security requirements document that was ultimately delivered to the client.

Step 8: Prioritize Requirements

In the first case study, the nine security requirements were prioritized based on the following qualitative rankings:

- Essential: Product will be unacceptable absent these requirements.
- Conditional: Requirement would enhance safety and security, but the product would not be unacceptable in its absence.
- Optional: Requirement may or may not be necessary.

Recalling the requirements identified in Steps 6-7, Req 1, which dealt with authentication at borders and gateways, was deemed essential because of its importance in protecting against the authentication-related risks outlined as a major risk in the risk assessment. Req 3, dealing with continuity of operations planning, is still seen as an important element and worth considering, but was found to be an optional requirement relative to the other eight requirements. That is, though COOP plans are valuable, the risk assessment phase found that the greater threats to the system were those that dealt with unauthorized disclosure of information, rather than availability attacks.

Another case study team utilized the Analytic Hierarchy Process (AHP) methodology to prioritize

requirements and found it to be very successful both in client acceptance and in its ability to handle security requirements [Karlsson 97, Saaty 80]. AHP is a technique for decision making in situations in which multiple objectives are present. The method calculates the relative value and cost among security requirements. By using AHP, the requirements engineer can also confirm the consistency of the stakeholders' results, which can prevent subjective judgment errors and increase the likelihood that the results are more reliable. The stakeholders found AHP valuable not only for its ability to quickly prioritize the security requirements but also because of the internal discussion that is stimulated.

Step 9: Requirements Inspection

The case study teams experimented with different inspection techniques and had different levels of success with each. None of the inspection techniques that were used were sufficiently effective in identifying defects in the security requirements, and the teams do not recommend their use in the future. Instead, the teams recommend that future iterations of SQUARE experiment with the Fagan inspection technique, which is a highly structured and proven technique for requirements inspection.

In one case study instance, each team member played a role in inspecting the quality of the team's work and deliverables. A peer review log was created to document what had been reviewed and was used to maintain a log of all problems, defects, and concerns. Each entry in the log was numbered and dated, addressing the date, origin, defect type, description, severity, owner, reviewer, and status. Each piece of documentation was assigned to an owner, who was held responsible for making sure that defects were fixed. This step was used as a sanity check to ensure that the team's work met the group's quality goals and expectations.

Managing and Assessing SQUARE

The final output to the client was a security requirements document that began by addressing the business goal, followed by the three security goals that supported this business goal, the nine categorized security requirements that supported the higher level security goals, and a list of application- and configuration-specific recommendations to meet these security requirements. From here, a responsible firm would use this document in the early stages of the development life cycle to make sure that security requirements are built into the planning of the project. Once a system has been deployed, the firm can look back to its requirements documentation to analyze whether it meets its requirements and thus satisfies its security goals to protect the system's business function. As change occurs—be it a configuration concern in the system, the organization's risk profile, or overall business goal—the process can be reused to plan how the changing environment will affect the security concerns of the system. SQUARE is thus easily reapplied to a system as needed.

Because the key players include a dedicated task force with knowledge of security who team with a group of knowledgeable client personnel, conducting a SQUARE assessment only requires that a firm have the time and human resources available to assist a group of outside analysts. Further, a firm knowledgeable in security could be in a position to conduct SQUARE analysis without outside help. The first graduate team spent a significant amount of time with the client in helping the client develop documentation. Many firms may complete this step before the SQUARE analysis begins. The second phase team made use of this documentation and was able to complete its assessment with very little client/analyst interaction. The SQUARE analysis was very lightweight and unobtrusive to the client in this regard. The third team worked with the initial client and two other clients, focusing on Steps 5 through 9.

Glossary

Fagan Inspection	A formal inspection method developed by Michael Fagan. http://www.mfagan.com/process.html

JAD	Joint Application Development—a focused workshop. http://www.credata.com/research/jad.html
SWEBOK	Software Engineering Body of Knowledge. http://www.swebok.org

References

[Alberts 04] Alberts, C.; Dorofee, A.; & Woody, C. "Considering Operational Security Risks During Systems Development." *SEPG 2004* (CD-ROM). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004.

[Butler 02] Butler, Shawn. "Security Attribute Evaluation Method: A Cost-Benefit Approach," 232-240. *Proceedings of the 24th International Conference on Software Engineering*. Orlando, FL, May 19-25, 2002. New York, NY: ACM Press, 2002.

[Checkland 89] Checkland, Peter. *Soft Systems Methodology. Rational Analysis for a Problematic World*. New York, NY: John Wiley & Sons, 1989.

[Chen 04] Chen, P.; Dean, M.; Ojoko-Adams, D.; Osman, H.; Lopez, L.; & Xie, N. *Systems Quality Requirements Engineering (SQUARE) Methodology: Case Study on Asset Management System* (CMU/SEI-2004-SR-015). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/publications/documents/04.reports/04sr0

[Cornford 04] Cornford, Steven L.; Feather, Martin S.; & Hicks, Kenneth A. *DDP – A Tool for Life-Cycle Risk Management*. http://ddptool.jpl.nasa.gov/docs/f344d-slc.pdf (2004).

[GAO 99] U.S. General Accounting Office. "Information Security Risk Assessment: Practices of Leading Organizations, A Supplement to GAO's May 1998 Executive Guide on Information Security Management." Washington, D.C.: U.S. General Accounting Office, 1999.

[Gordon 05] Gordon, D.; Mead, N. R.; Stehney, T.; Wattas, N.; & Yu, E. *System Quality Requirements Engineering (SQUARE): Case Study on Asset Management System, Phase II* (CMU/SEI-2005-SR-005). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2005.

http://www.sei.cmu.edu/publications/documents/05.reports/05sr0

[Haimes 04] Haimes, Yacov Y. *Risk Modeling, Assessment, and Management,* 2nd ed. Hoboken, NJ: John Wiley and Sons, Inc., 2004.

[Hubbard 99] Hubbard, R. "Design, Implementation, and Evaluation of a Process to Structure the Collection of Software Project Requirements." PhD diss., Colorado Technical University, 1999.

[Jacobson 92] Jacobson, Ivar. *Object-Oriented Software Engineering: A Use Case Driven Approach*. Boston, MA: Addison-Wesley, 1992.

[Kang 90] Kang, K.; Cohen, S.; Hess, J.; Novak, W.; & Peterson, A. Feature-Oriented Domain Analysis (FODA) Feasibility Study (CMU/SEI-90-TR-021, ADA235785). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1990. http://www.sei.cmu.edu/publications/documents/90.reports/90.tr.

Karlsson 97] Karlsson, J. & Ryan K. "A Cost-Value Approach for Prioritizing Requirements." *IEEE Software 14*, 5 (Sept./Oct. 1997): 67-74.

[Kunz 70] Kunz, Werner & Rittel, Horst. "Issues as Elements of Information Systems." http://www-iurd.ced.berkeley.edu/pub/WP-131.pdf (1970).

[Lipson 01] Lipson, Howard F.; Mead, Nancy R.; & Moore, Andrew P. *A Risk-Management Approach to the Design of Survivable COTS-Based Systems*. http://www.cert.org/research/isw/isw2001/papers/Lipson-29-08-a (2001).

[Mead 02] Mead, N. R. *Survivable Systems Analysis Method*. http://www.cert.org/archive/html/analysis-method.html (2002).

[Mead 04] Mead, N. R. "Requirements Elicitation and Analysis Processes for Safety & Security Requirements." *Proceedings of the Third International Workshop on Requirements for High Assurance Systems (RHAS 2004)*. Kyoto, Japan, Sept. 6, 2004. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/community/rhas-workshop/rhas04-proce

[Mead 05a] Mead, N.R., Hough, E. & Stehney, T. *Security Quality Requirements Engineering (SQUARE) Methodology*(CMU/SEI-2005-TR-009). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2005.

http://www.sei.cmu.edu/publications/documents/05.reports/05tr0

[Mead 05b] Mead, N. R. & Stehney, T. "Security Quality Requirements Engineering (SQUARE) Methodology." *Software Engineering for Secure Systems (SESS05), ICSE 2005 International Workshop on Requirements for High Assurance Systems*. St. Louis, MO, May 15-16, 2005.

[Mullery 79] Mullery, G. P. "CORE: A Method for Controlled Requirements Specification." *Proceedings of the 4th International Conference on Software Engineering*. Los Alamitos, CA: IEEE Computer Society Press, 1979.

[NSA 04] National Security Agency. *INFOSEC Assessment Methodology*. http://www.iatrp.com/iam.cfm (2004).

[QFD 05] QFD Institute. *Frequently Asked Questions About QFD*. http://www.qfdi.org/what_is_qfd/faqs_about_qfd.htm (2005).

[Saaty 80] Saaty, T. L. *The Analytic Hierarchy Process*. New York, NY: McGraw-Hill, 1980.

[Schiffrin 94] Schiffrin, D. *Approaches to Discourse*. Oxford, England: Blackwell, 1994.

[SEI 04] Software Engineering Institute. *International Workshop on Requirements for High Assurance Systems*, Kyoto, Japan, Sept. 6, 2004. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/community/rhas-workshop/rhas04-proce

[Stoneburner 02] Stoneburner, Gary; Goguen, Alice; & Feringa, Alexis. *Risk Management Guide for Information Technology Systems* (Special Publication 800-30). Gaithersburg, MD: National Institute of Standards and Technology, 2002. http://csrc.nist.gov/publications/nistpubs/800-30/sp800-30.pdf.

[Wood 89] Wood, Jane & Silver, Denise. *Joint Application Design: How to Design Quality Systems in 40% Less Time*. New York, NY: John Wiley & Sons, 1989.

[Woody 04] Woody, Carol; Hall, Anthony; & Clark, John. "Can Secure Systems be Built Using Today's Development Processes?" Panel presentation, European SEPG, London, England, June 17, 2004. http://www.cert.org/archive/pdf/eursepg04.pdf.

[Woody 05] Woody, C. *Eliciting and Analyzing Quality Requirements: Management Influences on*

Software Quality Requirements (CMU/SEI-2005-TN-010). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/publications/documents/05.reports/05tn0

[Xie 04] Xie, Nick & Mead, Nancy R. *SQUARE Project: Cost/Benefit Analysis Framework for Information Security Improvement Projects in Small Companies* (CMU/SEI-2004-TN-045). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/publications/documents/04.reports/04tn0

SEI Copyright

####

###	########
Copyright Holder	SEI

####

###	########
is-content-area-overview	false
Content Areas	Best Practices/Requirements Engineering
SDLC Relevance	Requirements
Workflow State	Publishable

1. http://www.sei.cmu.edu/about/legal-permissions.html